Stock market direction prediction for apple using Alpha 101 & Classification algorithm

Sanjay Kumar Tiwary . Sandeep R Diddi . Joyeeta Mallik . Sasirekha Sathasivam

1. Abstract :

In the world of finance, stock trading is one of the most important activities. Professional traders have developed a variety of analysis methods such as fundamental analysis, technical analysis, quantitative analysis, and so on. Such analytically methods make use of different sources ranging from news to price data, but they all aim at predicting the company's future stock prices so they can make educated decisions on their trading.

We prepare have imported dataset from yahoo finance and features data from alpha 101 and few calculations using artificial intelligence techniques to predict stock direction. Here the direction of stock and our prediction signal i.e. alpha should match to give good prediction.. This paper can help in predicting the stock market share like google direction to help in building trading strategy. With the help of this algorithm user will be able to perform sell and buy correctly.

2. Summary of problem statement, data and findings

In recent years, the increasing prominence of machine learning in various industries have enlightened many traders to apply machine learning techniques to the field, and some of them have produced quite promising results. In this paper, we will focus on short-term price prediction on general stock using time series data of stock price.

Predicting trends in stock market prices has been an area of interest for data scientist & researchers for many years due to dynamic nature. Volatility in stock market makes the task of prediction of stock direction is challenging.

Market risk, strongly correlated with forecasting errors, needs to be minimized to ensure minimal risk in investment. This paper is the outcome of experiments with a different approach. We just try to predict whether prices will increase or decrease. The problem is posed as a classification problem, where the class labels may be -1, +1, indicating an increase or a decrease in the price of a stock with respect to n days back.

For this purpose, the potential of Random Forests and KNN. Random Forests use an ensemble of Decision Trees to improve the accuracy of classification. Technical indicators such as Alpha 101, rolling average is used as features to train the model. The algorithms are shown to outperform the algorithms used in the existing literature.

The data selected for Apple stock for duration of five years, sample dataset is shown below

Date	Open	High	Low	Close	Adj Close	Volume
2/3/2014	71.80	72.53	71.33	71.65	60.98	100366000
2/4/2014	72.26	72.78	71.82	72.68	61.86	94170300
2/5/2014	72.37	73.61	72.32	73.23	62.32	82086200
2/6/2014	72.87	73.36	72.54	73.22	65.02	64441300

2/7/2014	74.48	74.70	73.91	74.24	65.93	92570100
2/10/2014	74.09	76.00	74.00	75.57	67.11	86389800

Table 1: Apple stock price (Raw data)

Statistics	Open	High	Low	Close	Volume
count	1259.0	1259.0	1259.0	1259.0	1259
mean	132.5	133.7	131.4	132.6	41839490
std	36.9	37.2	36.5	36.8	21504980
min	71.8	72.5	71.3	71.6	11475900
25%	105.5	106.5	104.8	105.7	26518500
50%	121.1	122.2	120.3	121.3	36379100
75%	159.0	160.1	157.6	158.7	51141350
max	230.8	233.5	229.8	232.1	189977900

Table 2: Basic Statistics of stock price

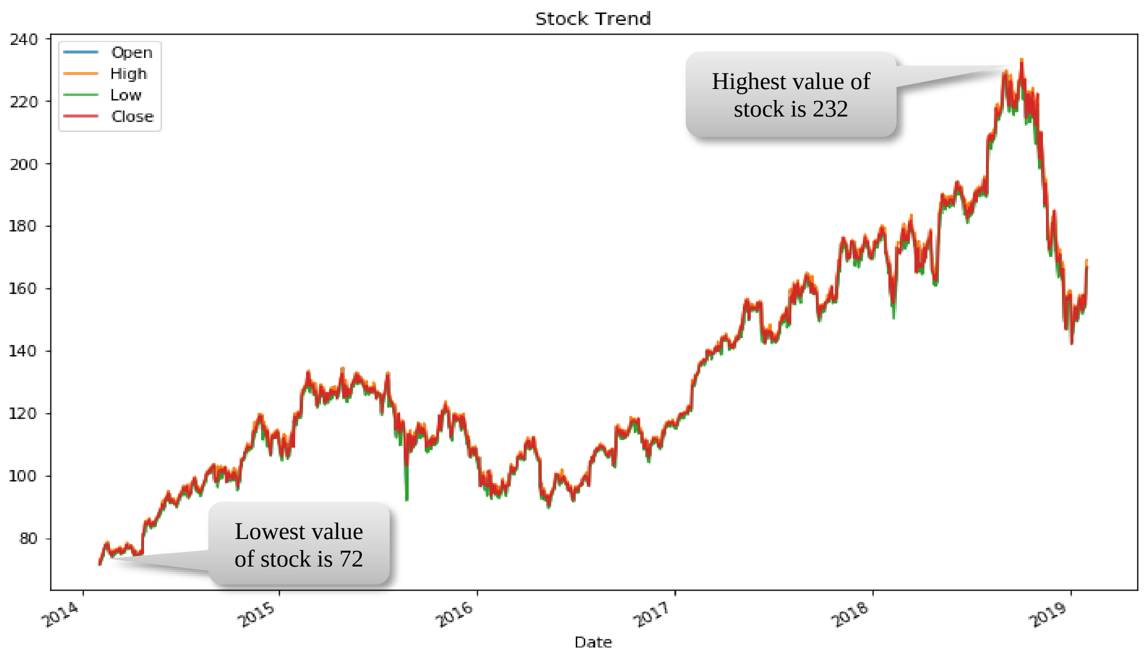


Figure 1: Trend of 5 year stock price

3. Overview of the final process

In our experiments, the time series data acquired is first exponentially smoothed. Then the technical indicators are extracted. Technical indicators provide insights to the expected stock price behaviour in future. These technical indicators are then used as features to train the classifiers. The indicators used in the current work will be discussed in this section.

Exponential smoothing grants larger weights to the recent observations and exponentially decreases weights of the past observations. The exponentially smoothed statistic of a series Y can be recursively calculated as:

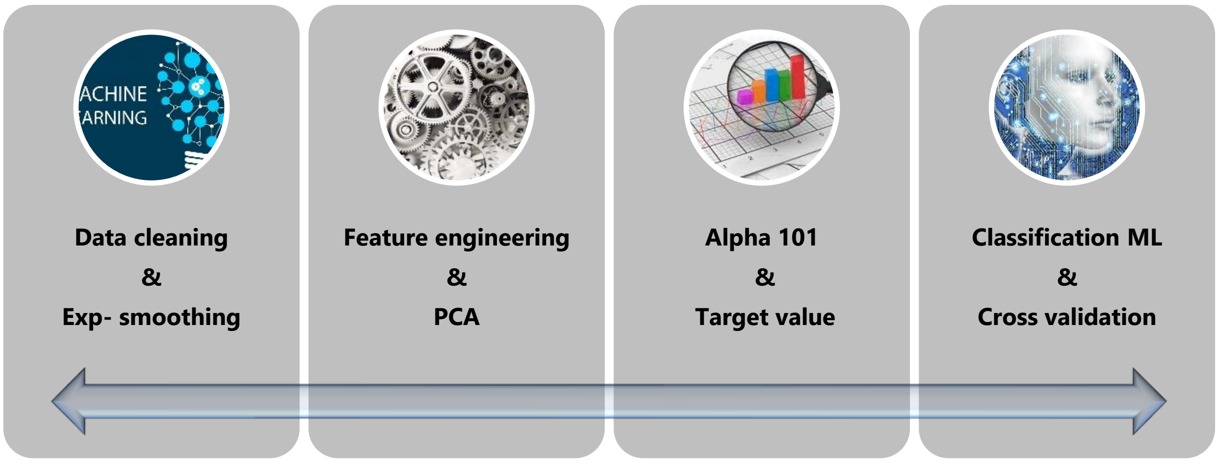


Figure 2: Framework of supervised learning in the current work

Calculation of alphas, there are sevral alphas and we have selected Alpha 101 for our price predictions.

In this section we describe some general features of our 101 formulaic alphas. The alphas, as are proprietary to WorldQuant LLC and are used here with its express permission. We provide as many details as we possibly can within the constraints imposed by the proprietary nature of the alphas. The formulaic expressions

Formula 1: **Alpha#101**: ((close - open) / ((high - low) + .001))

The target label or Y of the machine learning model is created based on this formula. When the value of target is +1, it indicates that there is a positive shift in the price, -1 indicates that there is a negative shift, giving us an idea of the direction of the prices for the respective stock. The target values are assigned as labels to the feature matrix.

Formula 2: **target** = sign(Today close price - Yesterday close price)

4. Solution overview

The problem of identifying the stock price direction is hard and requires lot of computation. We have first identified the output/ label for our scenario. This is the pattern of todays close price from yesterday close price. Then after this many step for feature engineering are taken. Like development of rolling average of 5 days, 10 days, 15 days and calculation of alpha 101. For this classification problem, we implemented SVM with rbf kernel, K-NN, Random forest in the sklearn library and ran them on the prices of one specific stock named 'APPLE'.

We have started with Stacking algorithm to find the best algorithm for this case and overall recall and accuracy. Then after that Principle component analysis (PCA) is applied on the no of column as all are corelated with each other, as shown in heat map. After PCA, again applied the Stacking classifier.

Then we applied Support Vector Regression (SVR) with radial basis kernel. Although this model performs really well, we cannot run it on entire data set due to our limited computing power, ad specially in production the deployment is very difficult.

Then K-Nearest neighbours is working good and Random forest, using some regularisation parameter and we have also applied the grid search CV to find the best params.

Then for model comparison, K- fold validation has worked superb and plotted the output using box plot to find the mean of results and its overall accuracy.

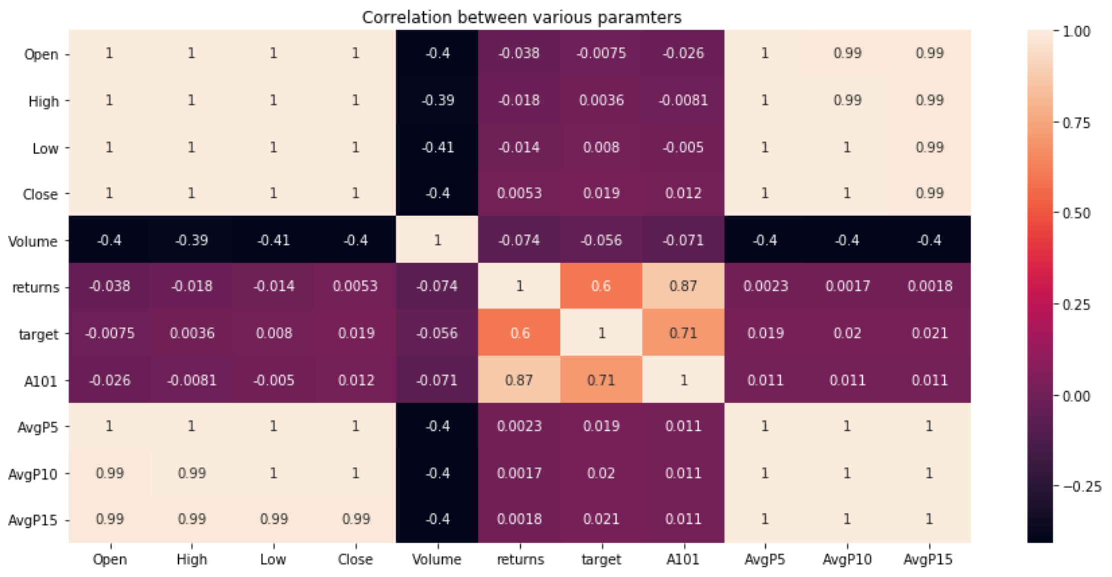
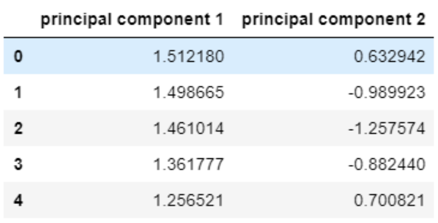


Figure 3: Heatmap of stock parameters (correlation)

Principal component analysis (PCA):

We have applied PCA on all the feature for dimension reduction. PCA uses linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower dimensional space.

PCA is applied to reduce the dimensions of the data set and the reason for this is as per Linear Algebra if the prediction be showcased on 2 dimensional then can be easily shifted to N-Dimensions by doing PCA we are calculating eigen values for eigen vectors. Every pair of the eigen vector are perpendicular to each other and hence are termed to be 0 because the cos 90 is 0



	principal component 1	principal component 2
0	1.512180	0.632942
1	1.498665	-0.989923
2	1.461014	-1.257574
3	1.361777	-0.882440
4	1.256521	0.700821

Table 3: Principal component

This is final data frame which is going to be pass into model, feature extraction

Date	Open	High	Low	Close	Volume	returns	target	A101	AvgP5	AvgP10	AvgP15
1/11/2019	152.4	154.0	151.3	152.8	27023200.0	0.0	-1	0.150	154.339	155.387	158.401
1/10/2019	152.4	153.8	151.1	152.9	35780700.0	0.0	-1	0.190	153.842	154.769	157.493
1/9/2019	152.1	154.0	150.7	153.0	45099100.0	0.0	1	0.281	153.381	154.355	156.596
1/8/2019	151.4	153.5	150.0	152.5	41025300.0	0.0	1	0.319	152.928	154.016	155.690
1/7/2019	150.5	152.2	148.8	151.2	54777800.0	0.0	1	0.200	152.487	153.647	154.921

Table 4: Final Data frame for model development

5. Model evaluation

5.1. Stacking algorithms

Stacking is an ensemble learning technique to combine multiple classification models via a meta-classifier. The individual classification models are trained based on the complete training set; then, the meta-classifier is fitted based on the outputs meta-features of the individual classification models in the ensemble. The meta-classifier can either be trained on the predicted class labels or probabilities from the ensemble.

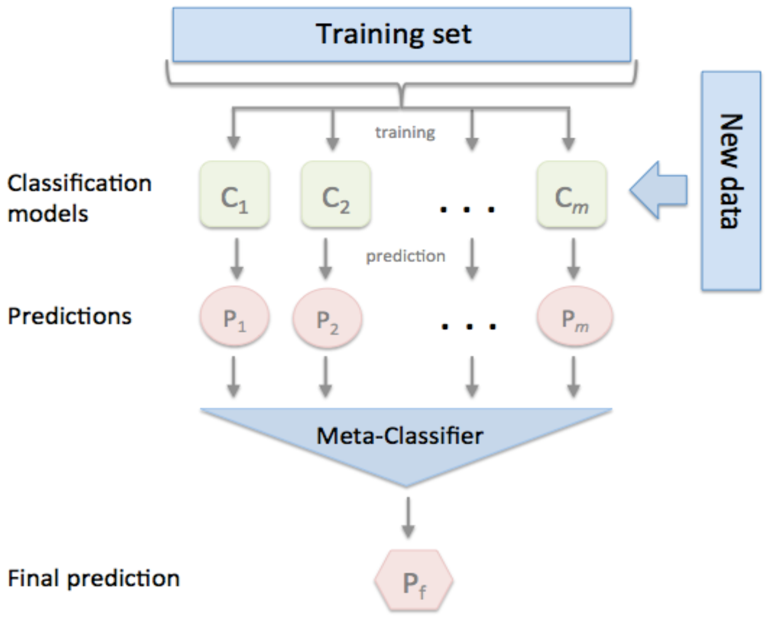


Figure 4: Stacking Classifier this link for refreance
(http://rasbt.github.io/mlxtend/user_guide/classifier/StackingClassifier_files/stackingclassification_overview.png)

Output of stacking Algorithm:

3-fold cross validation:

- Accuracy: 0.83 (+/- 0.03) [KNN]
- Accuracy: 0.81 (+/- 0.04) [Random Forest]
- Accuracy: 0.83 (+/- 0.03) [Support vector]
- Accuracy: 0.81 (+/- 0.03) [Stacking Classifier]

5.2. K- Nearest Neighbors:

Brief: The KNN algorithm is a robust and multipurpose classifier, that is often used as a benchmark for classifiers such as Artificial Neural Networks (ANN) and Support Vector Machines (SVM). KNN is having edge over many other classifier and is used in several of applications like economic forecasting, data compression and Health care.

KNN is categorized under **supervised learning** family of algorithms. It means we are assigning a labelled dataset for training observations and will be finding the relationship between them. Our goal is find the pattern hidden inside an unseen observation and then we can firmly predict the real case data in production.

Working philosophy : the K-nearest neighbour algorithm finding a majority vote between the K most similar instances to a given dataset. Similarity is calculated to a distance metric between two data points. Euclidean distance given,

$$d(\mathbf{p},\mathbf{q}) = d(\mathbf{q},\mathbf{p}) = \sqrt{(q_1-p_1)^2 + (q_2-p_2)^2 + \cdots + (q_n-p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}.$$

https://en.wikipedia.org/wiki/Euclidean_distance

Pros:

Very easy to understand and implement: A k-NN implementation does not require such code and can be a quick and simple way to begin machine learning datasets.

Does not assume any probability distributions on the input data. This can come in handy for inputs where the probability distribution is unknown and is therefore robust.

Cons:

Sensitive to localized data: Since k-NN gets all of its information from the input's neighbours, localized anomalies affect outcomes significantly, rather than for an algorithm that uses a generalized view of the data.

Computation time: Lazy learning requires that most of k-NN's computation be done during testing, rather than during training. This can be an issue for large datasets.

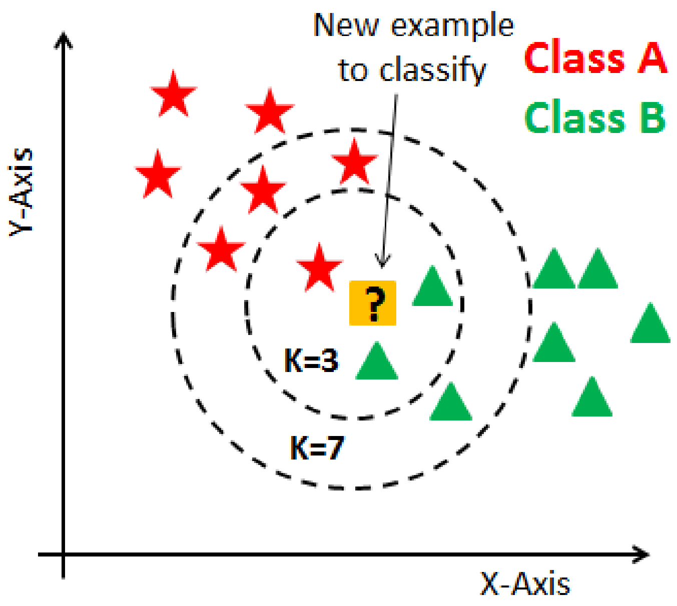


Figure 5: KNN reference
(http://res.cloudinary.com/dyd911kmh/image/upload/f_auto,q_auto:best/v1531424125/Knn_k1_z96jba.png)

Output of KNN

- K Nearest Neighbors (NN = 20)
- Accuracy Score: 85.8%
- Recall : 86 %
- Precision: 86 %

5.3. Random forest classifier:

Brief: Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. Random decision forests correct for decision trees habit of overfitting to their training set.

RF is used for ensemble of decision trees. It uses base principle of bagging with random feature selection to create more diverse trees. Splitting a node during the construction of a tree, the split that is chosen is no longer the best split among all the features. Instead, the split is picked is the best split among a random subset of the features

As a result of this randomness, the bias of the forest usually slightly increases (with respect to the bias of a single non-random tree)

Due to averaging, its variance decreases, usually more than compensating the increase in bias, hence yielding overall a better result can handle curse of dimensionality as the ensemble uses only a small random portion of the full feature set. Less prone to overfitting.

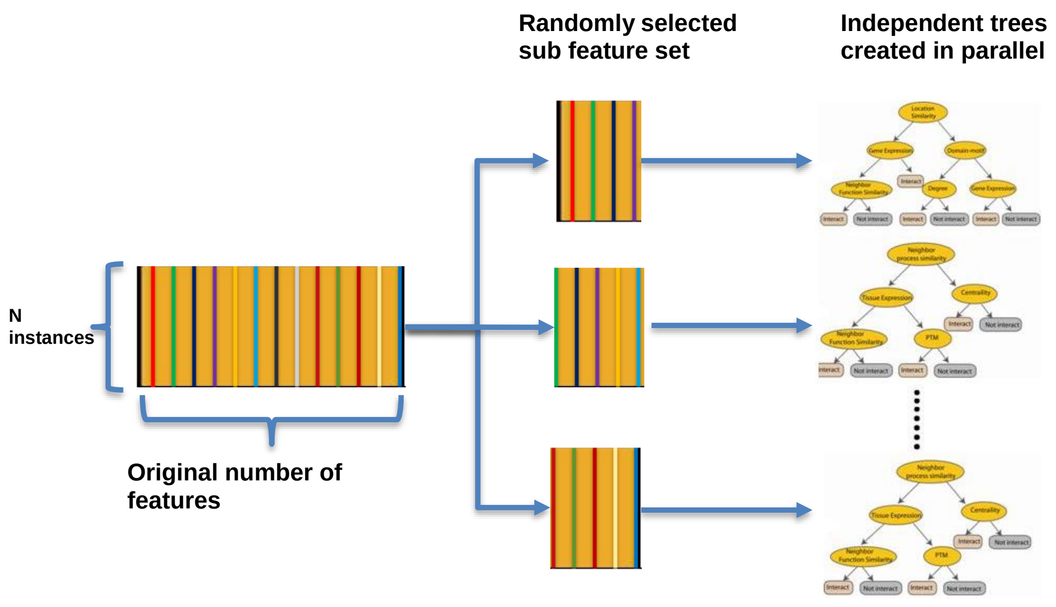

Randomly selected sub feature set
Independent trees created in parallel
N instances
Original number of features

Figure 6:Random forest,

Used with Decision Trees. Create different trees by providing different sub-features from the feature set to the tree creating algorithm. The optimization function is Entropy or Gini index.

Output of Random forest

- RF Accuracy Score: 86.08%
- Recall : 86%
- Precision: 86%

5.4. Support Vector Machine:

Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side

The kernel trick is a very intelligent mathematical technique that allows us to solve implicitly the linear separation problem in a higher dimensional feature space.

Kernel Trick:

SVM libraries come packaged with some standard kernel functions such as polynomial, radial basis function (RBF), and Sigmoid

a) For degree-d polynomials, the polynomial kernel looks like-

$$K(x,y) = (x^{\mathsf{T}} y + c)^d$$

where x and y are input vectors in lower dimension space, c is a user specified constant (usually 1). K denotes inner product of x, y in higher dimension space

b) RBF (Radial Basis Function) kernel on two samples x and x' is represented as -

$$K(\mathbf{x},\mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right)$$

It ranges from 0 when distance between x and x' increases (e^-infinity becomes 0) and becomes 1 when x = x' because x – x' = 0 and anything raised to 0 is 1

Strengths	Weakness
Very stable as it depends on the support vectors only.	Computationally intensive
Can be adapted to classification or numeric prediction problems	Prone to over fitting training data
Capable of modelling relatively more complex patterns than nearly any algorithm	Assumes linear relation between dependent and independent variables
No assumptions about underlying data sets	Generally treated as a black box model

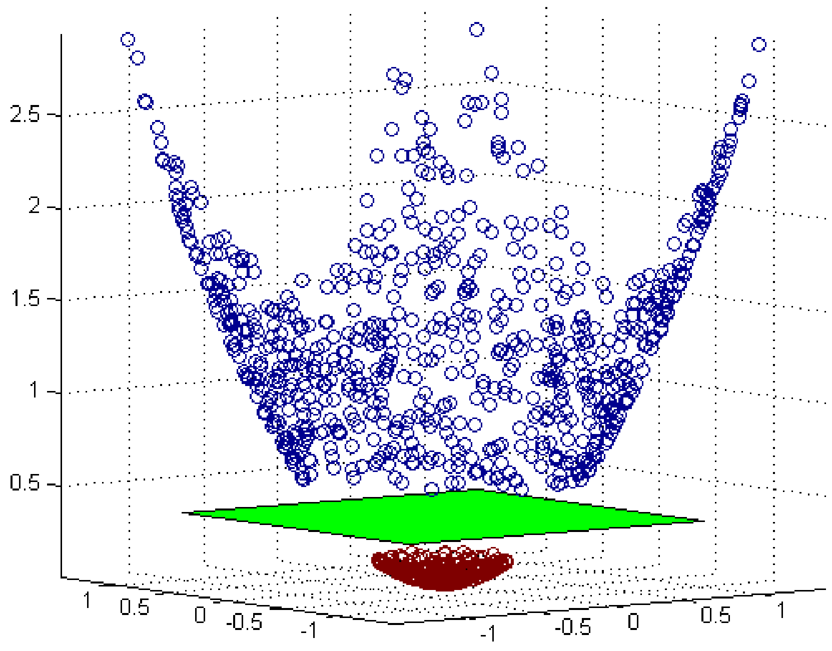


Figure 7: Support Vector Machine, reference
http://quantdare.com/wp-content/uploads/2016/09/svm_3d.png

Output of Support vector machine :

- Accuracy Score: 88%
- Recall: 88
- Precision: 88%

5.5. Model Comparison :

There are several method for this, for now table is manually prepared for all the three models and SVM is giving good best accuracy of 87 %. Also K-fold validation is applied and its box plot comparison is shown here.

Model	Accuracy	Precision	Recall
KNN	85%	86%	86%
SVM	88%	88%	88%
RF	86%	86%	86%

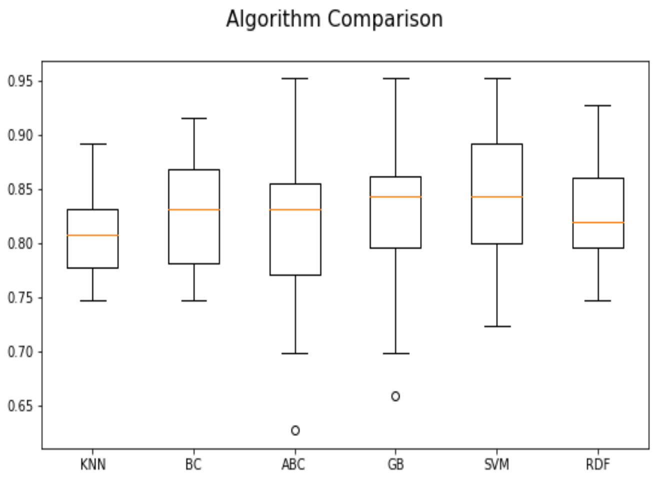


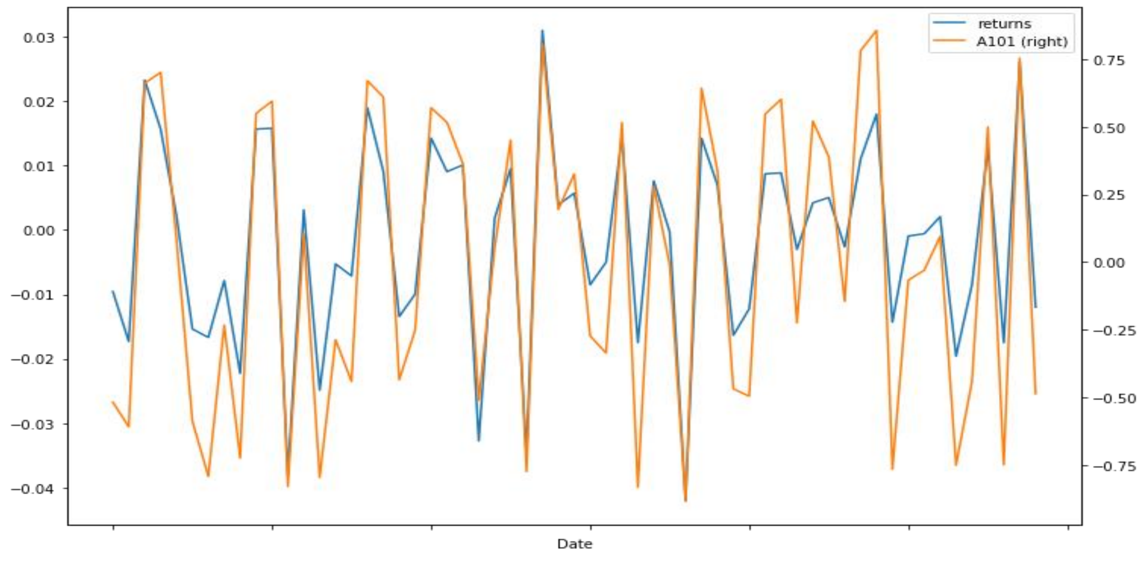
Figure 8: Algorithm comparison using Cross validation

6. Visualization(s)

This trend shows that alpha 101 and close price return are following each other in almost all the occasion during market going high or low.

Here Return is plotted on primary axis and prediction signal A101 is plotted in the secondary axis in the figure.

Figure 9: Returns vs Alpha 101



7. Limitations

As the world quant have 101 alpha available, but for academic purpose we have used only one alpha to calculate the signal prediction till end. The solution for the stock price prediction will be more effective if all the alpha in calculated and used as separate feature in machine learning models.

In this paper we have only used the Apple stock price for our analysis, but we can also use others market data and perform the similar analysis.

8. Closing Reflections

Stock market direction prediction very complex and uncertain due to its dynamics. However in the recent years, AI & machine learning technologies is now very effective in stock forecasting.
Many algorithms such as SVM, ANN etc. are used and applied for proveness in predicting stock direction. However, ensemble learning methods are still remained in top list of developers, we have used Random Forests to develop our predictive model and our model is giving very good accuracy and remarkable results. The model is found to be robust in predicting the direction of stock movement as the accuracy is above 80%.

The robustness of our model has been evaluated by calculating various parameters such as accuracy, precision, recall and F-score. For all the datasets we have used, we were able to achieve high accuracies for long-term Predicting the Direction of Stock Market Price Using Support vector and K-NN.

Our model can be used for building trading strategies for trading or to perform stock portfolio management, buy and sell strategy. This model shall be useful in to minimize the risk of investment in stock market by predicting the returns of a stock more accurately. Ensembles of different machine learning algorithms can also be checked for its robustness in stock prediction. We also recommend exploration of the application of Deep Learning practices in Stock Forecasting

The alpha used is i.e. Alpha 101 is real-life trading alphas used in production. This Alpha signal is giving very good accuracy and guidance for direction prediction.

Support vector machine (SVM) with kernel trick using Radial basis function gives the best accuracy of 88 %, but for production we recommend to use KNN or Random forest. As the SVM use lot of computational hardware space and may give slow output.

Our sample code is available at GitHub https://github.com/SandeepRDiddi/capstoneproject.

9. References

[1] Alice Zheng, Stanford University & Jack Jin, Stanford University
"Using AI to Make Predictions on Stock Market"

[2] Suryoday Basak , Snehanshu Saha, Saibal Kar, Luckyson Khaidem, Sudeepa Roy Dey
"*Predicting the Direction of Stock Market Price Using Tree Based Classifiers"*

[3] Zura Kakushadze, Geoffrey Lauprete , David Agmashenebeli Alley, WorldQuant LLC
"101 Formulaic Alphas" 2015.

[4] https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/